



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/759,737	01/16/2004	Jeffrey P. Snover	MS1-1902US	6905
22801	7590	05/18/2007		
LEE & HAYES PLLC 421 W RIVERSIDE AVENUE SUITE 500 SPOKANE, WA 99201			EXAMINER ZHEN, LI B	
			ART UNIT	PAPER NUMBER
			2194	
			NOTIFICATION DATE	DELIVERY MODE
			05/18/2007	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

lhptoms@leehayes.com

Office Action Summary

Application No.

10/759,737

Applicant(s)

SNOVER ET AL.

Examiner

Li B. Zhen

Art Unit

2194

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

- 1) ☒ Responsive to communication(s) filed on 28 February 2007.
- 2a) ☐ This action is **FINAL**. 2b) ☒ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

- 4) ☒ Claim(s) 1-27 and 29-32 is/are pending in the application.
- 4a) Of the above claim(s) _____ is/are withdrawn from consideration.
- 5) ☐ Claim(s) _____ is/are allowed.
- 6) ☒ Claim(s) 1-27 and 29-32 is/are rejected.
- 7) ☐ Claim(s) _____ is/are objected to.
- 8) ☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

- 9) ☐ The specification is objected to by the Examiner.
- 10) ☐ The drawing(s) filed on _____ is/are: a) ☐ accepted or b) ☐ objected to by the Examiner.
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) ☐ All b) ☐ Some * c) ☐ None of:
- ☐ Certified copies of the priority documents have been received.
 - ☐ Certified copies of the priority documents have been received in Application No. _____.
 - ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

- 1) ☒ Notice of References Cited (PTO-892)
- 2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
- 3) ☒ Information Disclosure Statement(s) (PTO/SB/08)
Paper No(s)/Mail Date 1/8/2007

- 4) ☐ Interview Summary (PTO-413)
Paper No(s)/Mail Date. _____
- 5) ☐ Notice of Informal Patent Application
- 6) ☐ Other: _____

DETAILED ACTION

1. Claims 1 – 27 and 29 – 32 are pending in the application.

Response to Arguments

2. In response to the Non-Final Office Action dated 09/07/2006, applicant argues:

(1) Russell discloses a mechanism for dynamically creating a customizable serializer which is derived from some existing "template" or "default constructor" [paragraph 0015];

(2) Russell creates its customized serializer in response to a failure of a target object (i.e., instance of an object) to match its predefined definition for its object type. For example, Russell uses traditional reflection (which is described briefly in App, p. 18, lines 17-18) [paragraph 0016]; and

(3) For Russell to function properly there must be a default constructor that matches the properties in the target object [paragraph 0017].

In response to argument (1), examiner respectfully disagrees and submits that the customizable serialized of Russell does is not derived from existing template or default constructor. Russell discloses customizing type mapping dynamically ("on the fly") [p. 3, paragraph 0050 of Russell]. The mapping, which corresponds to the applicant's claimed list of known types, is customized during runtime and not the serializer. Contrary to applicant's submission, Russell does not disclose a template or default constructor for a serializer. The default constructor in Russell is used to create an instance of a class that is part of the complex object [i.e., default constructor for the embedded objects inside a bean, pp. 5 - 6, paragraph 0072]. The default constructor is

used to ensure that the embedded objects of a complex object (i.e. a bean) are instantiated so that the embedded objects of the complex object can be serialized with the complex object.

As to argument (2), examiner respectfully disagrees and notes that Russell does not create a customized serializer in response to a failure of a target object [Note: Russell does not disclose a target object in the specification. It appears applicant is referring to the object that is being serialized as a target object] to match its predefined definition for its object type. Instead, Russell discloses generating a type mapping for a complex object when the serializer has not mapping information for the complex object [this is the same as applicant's claimed unknown object]. Russell then performs a recursive process to serialize the complex object by serializing the embedded objects inside the complex object [p. 4, paragraph 0057 and pp. 5 – 6, paragraph 0072]. Although Russell uses reflection, examiner notes that the currently presented claims do not preclude the use of reflection. In fact, applicant's specification discloses performing reflection on an object to determine the type of the object [At block 1006, the type of object is identified by reflecting on the object; p. 36, line 15 of applicant's specification].

As to argument (3), examiner notes that the claims do not preclude the use of a default constructor. The default constructor in Russell is used to create an instance of a class that is part of the complex object [i.e., default constructor for the embedded objects inside a bean, pp. 5 - 6, paragraph 0072]. The default constructor is used to ensure that the embedded objects of a complex object (i.e. a bean) are instantiated so that the embedded objects of the complex object can be serialized with the complex

Art Unit: 2194

object. In order to use reflection to serialize objects, the objects have to be instantiated (i.e. At block 1006, the type of object is identified by reflecting on the object; p. 36, line 15 of applicant's specification, emphasis added). Since applicant's serialization process requires calling reflection on an object, the objects in applicant's invention would also require a default constructor.

Claim Rejections - 35 USC § 102

3. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

4. **Claims 1 – 8, 11, 13, 14, 16, 17, 26, 27 and 29 – 32 are rejected under 35 U.S.C. 102(e) as being anticipated by U.S. Patent Application Publication No. 2004/0039964 to Russell et al. [hereinafter Russell, cited in the previous office action].**

5. As to claim 1, Russell teaches at least one computer-readable storage medium [p. 8, paragraph 0096] having computer executable instructions that provide a method for transferring computer-readable objects [p. 5, paragraph 0071] across a remote boundary [p. 7, paragraph 0088], the method comprising:

without relying on a pre-defined definition of a specified data type of an object [the serializer 220 of preferred embodiments not only does not require this input, but it generates the type mapping 210 "on the fly", as needed to carry out a serialization; p. 3, paragraph 0050], decomposing the object [complex object 500 that needs to be serialized; p. 5, paragraph 0063] into multiple sub-components [A child element 930 is generated for the top-level class of the object being serialized; p. 6, paragraph 0080] wherein the decomposing comprising extracting discernable properties [attributes; p. 6, paragraph 0080] and values for each sub-components [values of these attributes are also generated by parsing the exception and locating the class name; p. 6, paragraph 0075];

without relying on the pre-defined definition of the specified data type of the object [the serializer 220 of preferred embodiments not only does not require this input, but it generates the type mapping 210 "on the fly", as needed to carry out a serialization; p. 3, paragraph 0050], serializing the multiple sub-components and their discernable properties and values into a serialized package [entire complex object has been marshalled, the result is a serialized object as shown at 540; p. 4, paragraph 0057]; and

transmitting the serialized package to a remote entity [Message 1015 is then sent through the network 1020, 1025, and then reaches the target Web service 1030; p. 7, paragraphs 0087 and 0088 and p. 3, paragraph 0043],

wherein the decomposing, serializing, and transmitting facilities transferring computer-readable objects across a remote boundary [a JavaBean is serialized for

transmission to a Web service or other request message consumer; p. 4, paragraph 0059].

6. As to claim 26, this is a system claims that corresponds to product claim 1; note the rejection to claim 1 above, which also meets the limitations of this system claim. Russell also teaches the system comprising a processor [p. 8, paragraph 0097] and memory [p. 8, paragraph 0096].

7. As to claim 29, Russell teaches the decomposing act further comprises: dividing the multiple sub-components into a hierarchy based upon a list of known object types [A child element 930 is generated for the top-level class of the object being serialized; p. 6, paragraph 0080], the known object types being a type known by the remote entity [type mappings; p. paragraph 0077]; and associating an object with one of the known object types, so that object is a known object [serializer next needs to serialize an instance of AccountTransactions class, the type mappings will be available; p. 6, paragraph 0077].

8. As to claim 2, Russell teaches the list identifies the first type as one of the known object types [using selected ones of the stored type mappings when the serialization process is subsequently invoked for the determined object; p. 2, paragraph 0018 and p. 6, paragraph 0077].

9. As to claim 3, Russell teaches at least one sub-component comprises an unknown object having a type unidentified within the list [a "missing" type mapping that

causes an exception during serialization is programmatically resolved; p. 2, paragraph 0036].

10. As to claim 4, Russell teaches the decomposing an object further comprises decomposing the unknown object into another level of sub-components based on the list [A recursive process is then invoked to serialize an instance of this class. Once the entire complex object has been marshalled, the result is a serialized object as shown at 540; p. 4, paragraph 0057].

11. As to claim 5, Russell teaches a first process [object-based client code; p. 3, paragraph 0048] on a system transmits the serialized package [p. 4, paragraph 0057] and the remote entity comprises another process [Web service 1030; p. 7, paragraph 0088] on the system [client/server networks; p. 3, paragraph 0043].

12. As to claim 6, Russell teaches a first process [object-based client code; p. 3, paragraph 0048] on a system transmits the serialized package [p. 4, paragraph 0057] and the remote entity comprises another process [Web service 1030; p. 7, paragraph 0088] on another system [client/server networks; p. 3, paragraph 0043].

13. As to claim 7, Russell teaches a first application [object-based client code; p. 3, paragraph 0048] domain [client/server networks; p. 3, paragraph 0043] executing within a process transmits the serialized package [p. 4, paragraph 0057] and the remote entity comprises another application domain [client/server networks; p. 3, paragraph 0043] within the process [Web service 1030; p. 7, paragraph 0088].

14. As to claim 8, Russell teaches the hierarchy comprises a property bag ["mappings" element 800 contains a child "map" element for each mapping being generated for this JavaBean; p. 6, paragraph 0074].

15. As to claim 11, Russell teaches the property bag comprises a plurality of entries ["mappings" element 800 contains a child "map" element for each mapping; p. 6, paragraph 0074], each entry being associated with one of the sub-components [class name in the example is "AccountTransactions"; p. 6, paragraph 0075] and having a first field for storing a name ["qname" attribute; p. 6, paragraph 0075] associated with the sub-component [name of the class; p. 6, paragraph 0075], a second field for storing a value associated with the sub-component [values of these attributes are also generated by parsing the exception and locating the class name; p. 6, paragraph 0075], and a third field for storing a type associated with the sub-component [a "javaType" attribute; p. 6, paragraph 0075].

16. As to claim 13, Russell teaches negotiating the list by accepting a plurality of object types received from a first process [added to a mapping registry], the accepted object types becoming known object types identified within the list [programmatically-generated mappings are automatically added to a mapping registry; p. 2, paragraph 0037].

17. As to claim 14, Russell teaches negotiating the list by receiving an identifier for a file and having the list include object types identified within the file [encodingStyle"

attribute where that default value identifies the SOAP encoding schema; p. 6, paragraph 0074].

18. As to claim 16, Russell teaches limiting the hierarchy of sub-components [Each map element has six attributes; p. 6, paragraph 0074] by defining a property set that identifies individual properties of the object [attributes; p. 6, paragraph 0080], wherein decomposing the object comprises decomposing the identified individual properties of the object [For the object to be serialized, the values of a "qname" attribute and of a "javaType" attribute are programmatically generated; p. 6, paragraph 0075].

19. As to claim 17, Russell teaches limiting the hierarchy of sub-components by identifying a specified property within the object [a type attribute 932 is set to match the name of the class being serialized; p. 6, paragraph 0080], wherein decomposing the object comprises decomposing the specified property [element is created for the endDate field, as shown at 970, where this element has a type attribute 971; p. 7, paragraph 0085].

20. As to claim 27, this is a system claims that corresponds to product claim 5; note the rejection to claim 5 above, which also meets the limitations of this system claim.

21. As to claim 30, Russell teaches just the known objects are serialized in the serializing act [p. 4, paragraph 0057].

22. As to claim 31, Russell teaches the method further comprises negotiating with the remote entity to determine which object types are known by the remote entity [p. 2, paragraph 0037].

23. As to claim 32, Russell teaches negotiating with the remote entity to determine which object types are known by the remote entity [p. 2, paragraph 0037].

24. Claims 9, 10 and 12 are rejected under 35 U.S.C. 103(a) as being unpatentable over Russell in view of U.S. Patent Application Publication No. 2003/0182308 to Ernst et al. [hereinafter Ernst, cited in the previous office action].

25. As to claim 9, Russell teaches property bag [p. 6, paragraph 0074] but does not teach the property bag comprising a hash table.

However, Ernst teaches transforming data from one format to another [p. 5, paragraph 0056] based on data schema [p. 5, paragraph 0057], a property bag [a set of properties of that object; p. 4, paragraph 0039] comprising a hash table [lookup table; p. 4, paragraph 0039].

It would have been obvious to a person of ordinary skill in the art at the time the invention was made to combine the teachings of Ernst and Russell because Ernst's teachings allows content schema evolution while maintaining operation based on already stored content data [p. paragraph 0011 of Ernst] and simplifies maintenance and takes some burden of the setup phase [p. 10, paragraph 0118 of Ernst].

26. As to claim 10, Russell does not teach a key for each entry in the hash table comprises a name for the sub-component associated with the entry.

However, Ernst teaches a key [ID] for each entry in the hash table [selecting from the property tables all rows with the object ID associated to the content object; p. 11, paragraph 0132] comprises a name for the sub-component associated with the entry [properties are internally represented by identifiers; p. 11, paragraph 0128].

It would have been obvious to a person of ordinary skill in the art at the time the invention was made to combine the teachings of Ernst and Russell because Ernst's teachings provide unique invariable identifiers for the content types and/or properties [p. 5, paragraph 0051 of Ernst].

27. As to claim 12, Russell does not teach version numbers for an object type list.

However, Ernst teaches negotiating [pp. 4 – 5, paragraph 0047] the known object types identified within list by receiving a version number of a first list [label referring to a content schema version; p. 4, paragraph 0043] available to a first process [a label is used to record a set of versions that fulfill some interobject integrity constraints; p. 4, paragraph 0043], comparing the version number to another version number of a second list available to the remote entity [a version is directly linked from many sources, a modified version would only be linked after changing all referring resources; p. 4, paragraph 0044], and determining the list based on the comparison [generating new version of these resources; p. 4, paragraph 0044].

It would have been obvious to a person of ordinary skill in the art at the time the invention was made to combine the teachings of Ernst and Russell because Ernst's

Art Unit: 2194

teachings provide a unified approach to handle content schema evolution and content generation [p. 6, paragraph 0061 of Ernst].

28. Claims 15 and 18 – 25 are rejected under 35 U.S.C. 103(a) as being unpatentable over Russell in view of U.S. Patent Application Publication No. 2005/0154978 to Albornoz et al. [hereinafter Albornoz, cited in the previous office action].

29. As to claim 15, Russell does not teach limiting the hierarchy of sub-components by specifying a pre-determined depth for the hierarchy.

However, Albornoz teaches Serialization of the objects within an XML data structure API class [p. 3, paragraph 0031], hierarchical data structures with one or more layers of sub-elements [p. 4, paragraph 0043], limiting the hierarchy of sub-components by specifying a pre-determined depth for the hierarchy [a minimum number of elements in the array is not met and/or the maximum number of elements in the array is exceeded; p. 7, paragraph 0062], wherein decomposing the object comprises decomposing the object to the pre-determined depth [If the maxOccurs for the current sub-element is determined to not be greater than one; p. 7, paragraph 0067].

It would have been obvious to a person of ordinary skill in the art at the time the invention was made to combine the teachings of Albornoz and Russell because Albornoz's teaching raises an error if a maximum number of elements is exceeded or minimum number of elements is not met [p. 7, paragraph 0062 of Albornoz]. The raised

error would stop processing when the maximum number of elements is exceeded and this would prevent out of memory errors.

30. As to claim 18, Russell as modified by Albornoz teaches limiting the hierarchy of sub-components by specifying a pre-determined number [a minimum number of elements in the array is not met and/or the maximum number of elements in the array is exceeded; p. 7, paragraph 0062 of Albornoz] that limits the known objects that are serialized into the serialized package by the number [If the maxOccurs for the current sub-element is determined to not be greater than one; p. 7, paragraph 0067 of Albornoz]. As to the motivation for combining Russell and Albornoz, see the rejection to claim 15 above.

31. As to claim 19, Russell teaches the invention substantially as claimed including at least one storage computer-readable medium [p. 8, paragraph 0096] having computer executable instructions that provide a method for receiving a package representing a computer-readable object transmitted across a remote boundary [p. 7, paragraph 0088], the method comprising:

receiving a serialized package [serialize the objects to be passed as input to the Web service so that these objects and their data can be passed to the Web service; p. 3, paragraph 0048] from a remote entity [client code, p. 3, paragraph 0048];

identifying a hierarchy of sub-components ["mappings" element 800 contains a child "map" element for each mapping being generated for this JavaBean; p. 6,

paragraph 0074], the hierarchy representing an object of a first type [type mappings; p. paragraph 0077];

for each sub-component: identifying a type associated with the sub-component [mappings generated according to the present invention may be used when deserializing messages (i.e., when transforming flattened, or non-hierarchical, data structures back into their object format); p. 2, paragraph 0038];

determining whether that identified type is within a list of known object types [deserialization leverages the mappings that were previously created and stored in the mappings registry, as needed; p. 7, paragraph 0092];

responding to the determining, wherein the responding comprises instantiating an object of the type [deserialization process uses the stored type mappings to generate an object version of the message; p. 2, paragraph 0021], wherein the instantiating and populating are performed when the identified type is within the list of known object types [deserialization leverages the mappings that were previously created and stored in the mappings registry; p. 7, paragraph 0092]. Although Russell teaches the invention substantially, Russell does not specifically teach populating at least one property of the object with information obtained from within the serialized package.

However, Albornoz teaches serialization of the objects within an XML data structure API class [p. 3, paragraph 0031], hierarchical data structures with one or more layers of sub-elements [p. 4, paragraph 0043], instantiating an object of a type and populating at least one property of the object with information obtained from within the

serialized package [creates, at step 1004, an object that corresponds to the received schema; p. 7, paragraph 0065].

It would have been obvious to a person of ordinary skill in the art at the time the invention was made to combine the teachings of Albornoz and Russell because Albornoz's teaching provides easy manipulation within dynamic languages of data structures that reflect the structure of content models defined by XML schemas [pp. 5 – 6, paragraph 0053 of Albornoz].

32. As to claim 20, Russell teaches the list includes the first type as one of the known object types [deserialization leverages the mappings that were previously created and stored in the mappings registry; p. 7, paragraph 0092].

33. As to claim 21, Russell teaches the at least one sub-component comprises an unknown object having a type unidentified within the list [a "missing" type mapping; p. 2, paragraph 0036 and p. 4, paragraph 0059].

34. As to claim 22, Russell teaches a first process [Web service 1030; p. 7, paragraph 0088] on a system receives the serialized package [p. 4, paragraph 0057] and the remote entity comprises another process [object-based client code; p. 3, paragraph 0048] on the system [client/server; p. 3, paragraph 0043].

35. As to claim 23, Russell teaches a first process [Web service 1030; p. 7, paragraph 0088] on a system receives the serialized package [p. 4, paragraph 0057]

Art Unit: 2194

and the remote entity comprises another process [object-based client code; p. 3, paragraph 0048] on another system [client/server; p. 3, paragraph 0043].

36. As to claim 24, Russell teaches a first application domain [client/server networks; p. 3, paragraph 0043] executing within a process [Web service 1030; p. 7, paragraph 0088] receives the serialized package [p. 4, paragraph 0057] and the remote entity [object-based client code; p. 3, paragraph 0048] comprises another application domain [client/server networks; p. 3, paragraph 0043] within the process.

37. As to claim 25, Russell teaches the serialized package comprises an XML document [SOAP XML format; p. 4, paragraph 0057].

Conclusion

38. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

U.S. Patent No. 6,477,701 to Heistermann discloses adaptive serialization and deserialization processes based on the version of program objects.

CONTACT INFORMATION

39. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Li B. Zhen whose telephone number is (571) 272-3768. The examiner can normally be reached on Mon - Fri, 8:30am - 5pm.

Art Unit: 2194

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, William Thomson can be reached on 571-272-3718. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

Li B. Zhen
Examiner
Art Unit 2194

LBZ



5/11/2007